
InforWorkflow

Release 2.1

Daniel Jordan

2021-05-21

USAGE

1	Introduction	3
2	Installation	5
2.1	Docker Desktop	6
2.2	Prerequisites	6
2.3	Start on Mac or Linux	7
2.4	Start on Windows	7
2.5	Upgrade to Latest Build	7
3	Requirements	9
4	How to run	11
4.1	Install Docker on Windows	11
4.2	Install Docker on MacOS	13
4.3	How to create flows on Prefect cloud using Inforflow Docker instance	14
4.4	How to run flow from Prefect cloud	15
5	Configuration Files	19
6	Great Expectations Integration	23
6.1	Creating Expectation Suites	23
6.2	List Expectation Suites	23
6.3	Update Expectation Suites	24
7	Great Expectations Example	25
8	Common error messages	29
9	Indices and tables	31

Creates data migration flows

INTRODUCTION

InforFlow is a wrapper library to create **Prefect based flows** to migrate data **from MS-SQL Sever to Infor M3**.

In other words, it is using the Prefect library to create flows and uses the InforION library to tranform and load data to M3.

INSTALLATION

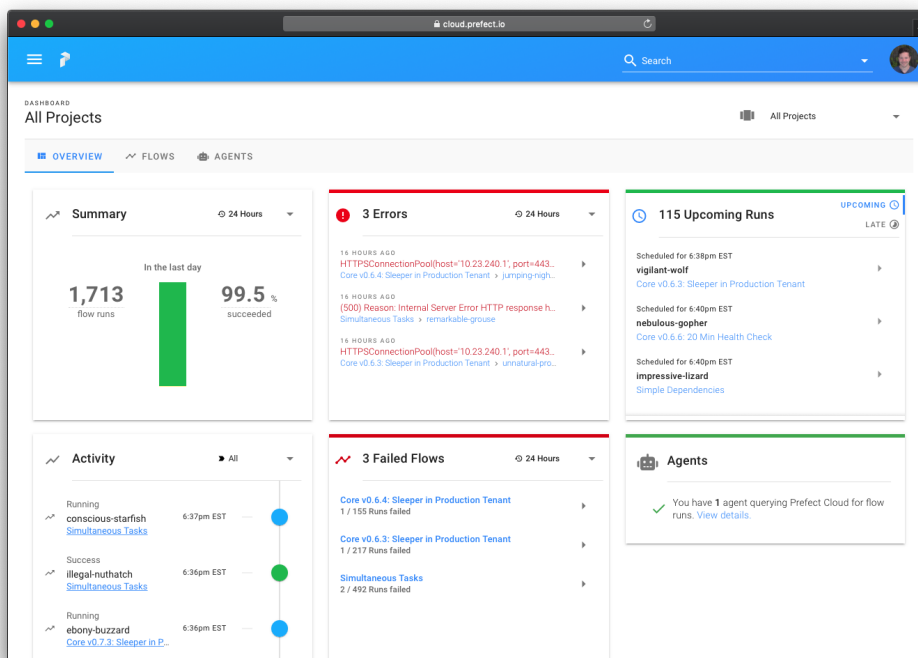
Firstly, Docker is an easy-to-install application for your Mac or Windows environment that can build and run packaged applications in containers and microservices on virtually any platform. In order to access and install the **inforion docker image**, so its available to use, please insert in your Terminal (MacOs) or Command Prompt (Windows):

```
docker run fellowconsulting/inforion
```

In any case, here you can access this link for more information about the inforion docker image:

[fellowconsulting/inforion Dockerhub](#)

A modern, *web-based UI* is provided to facilitate use, in which the various data flows (transformation to load) can also be scheduled. This is the **Prefect.io UI** where we display the Workflows. Here is a quick look:



2.1 Docker Desktop

Another important point, is to have downloaded **Docker Desktop**, which depending on your working environment, Mac or Windows, you can download it from here:

[Docker Desktop for Mac](#)

[Docker Desktop for Windows](#)

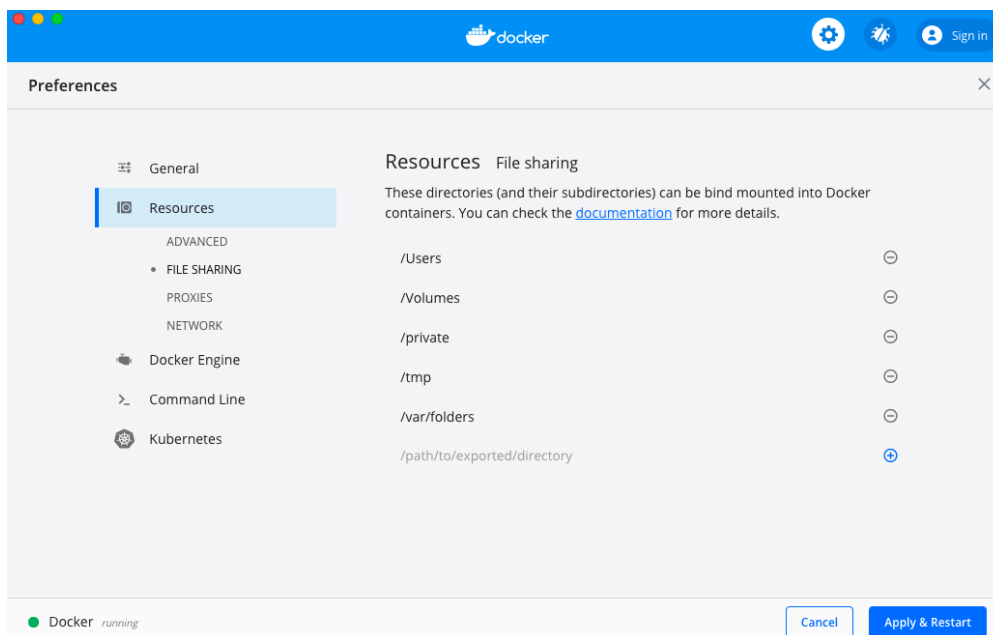
The important part about the **Docker Desktop** is that it allows you to access the required files to start running the prefect flows. That is why, firstly, you need to have access to the following files mentioned in “Prerequisites”.

2.2 Prerequisites

But, before you start, it is important that you check if you have created a **conf directory**. These are necessary files to make *test loads*, as well as for the data extraction, transformation and load. So, In the directory you should have the following files:

- **Mapping_Kundenstamm.xlsx**
- **credentials.yml**: Here you define the access to M3, the database with the respective data, the Prefect.io and Great Expectation environment.
- **IONAPI Key File**: Here the access to the APIs is provided.
- **module_dependencies.csv**: Here the dependencies between different objects are defined: Module_name and Parent_module. These are also displayed in the UI.
- **modules.csv**: Here parameters are defined that are used to display and run the dataflows in the UI.

Another important prerequisite is that we add the *conf folder* (as seen above) as well as the *Great Expectations folder* to the **Docker Desktop**. It is as simple as opening the Docker and getting into Resources, and next File Sharing, to add the new directories. When this is done, you can actually start testing your data loads with *Prefect flows*.



Just to finish this segment, here are some common and very important commands when starting and upgrading the inforion Docker image.

2.3 Start on Mac or Linux

```
docker run -v $PWD/conf:/ion-workflow/conf fellowconsulting/inforion flow -b /ion-  
↳workflow/conf -c credentials.yml -m modules.csv -d module_dependencies.csv
```

2.4 Start on Windows

```
docker run -v %cd%/conf:/ion-workflow/conf fellowconsulting/inforion flow -b /ion-  
↳workflow/conf -c credentials.yml -m modules.csv -d module_dependencies.csv
```

2.5 Upgrade to Latest Build

```
docker pull fellowconsulting/inforion:latest
```


REQUIREMENTS

You need following information and documents before you can create flows

- `Credentials`
 - **Access to MS-SQL Server**
 - * Server name
 - * Username
 - * Password
 - * Source Database name (Database from which data will be transfered to M3)
 - * Staging Database name (Database where data will store after transformation)
 - **Access to Prefect cloud**
 - * Token for Prefect cloud
 - * Project name
 - **Access to M3**
 - * M3 URL
 - * ION API File

All these credential informations should be provided in **credentials.yml**

- `Modules information`

For every module you want to export data from MS-SQL Server to M3 you need following information and add this to a csv file. Ideally it should be named **modules.csv**.

- Module name (You can put any name here. This is only used to create flows)
- Program name (This should be valid M3 program name where data will be exported)
- Mapping excel file path (Full path for the excel file which contains the sheet used for data tranformation)
- Sheet name (Name of the sheet which contains all the transformation information)
- Source table name: Table name from which data will be extracted.
- Stage table name: Table name where data will stored after transformation.

- `Modules dependencies information`

You should know if a module you are exporting already depends on another module. e.g. `OrderLineItems` is dependent on `Orders` and should be transferred after `Orders` are transferred. This

dependencies need to be provided in a csv file in a parent-child format, ideally named **module_dependencies.csv**

HOW TO RUN

InforFlow is available as a Docker instance.

Docker is a platform for running containers with prepackaged applications. It's a system that we are going to use for packaging and deploying different versions and microservices.

4.1 Install Docker on Windows

Install Docker Desktop on Windows.

System Requirements:

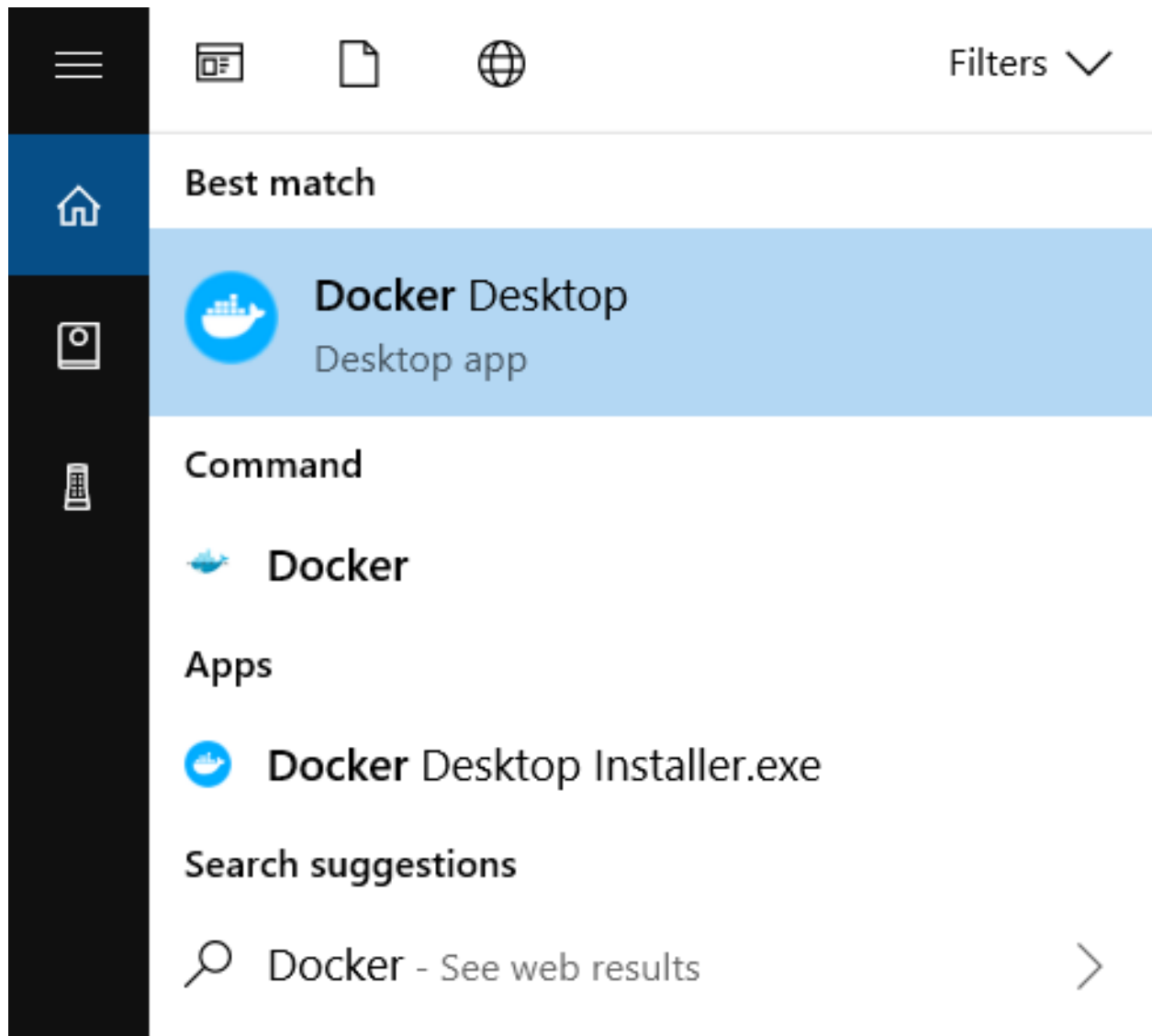
- Windows 10 64-bit: Pro, Enterprise, or Education (Build 16299 or later).
- Hyper-V and Containers Windows features must be enabled.
- The following hardware prerequisites are required to successfully run Client Hyper-V on Windows 10:
 - 64 bit processor with Second Level Address Translation (SLAT)
 - 4GB system RAM
 - BIOS-level hardware virtualization support must be enabled in the BIOS settings. For more information, see Virtualization.

Installation:

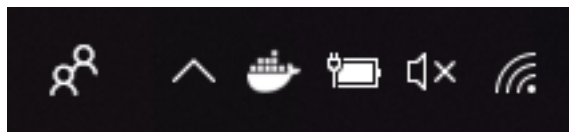
1. Double-click Docker Desktop Installer.exe to run the installer.
2. When prompted, ensure the Enable Hyper-V Windows Features option is selected on the Configuration page.
3. Follow the instructions on the installation wizard to authorize the installer and proceed with the install.
4. When the installation is successful, click Close to complete the process.

Start Docker Desktop:

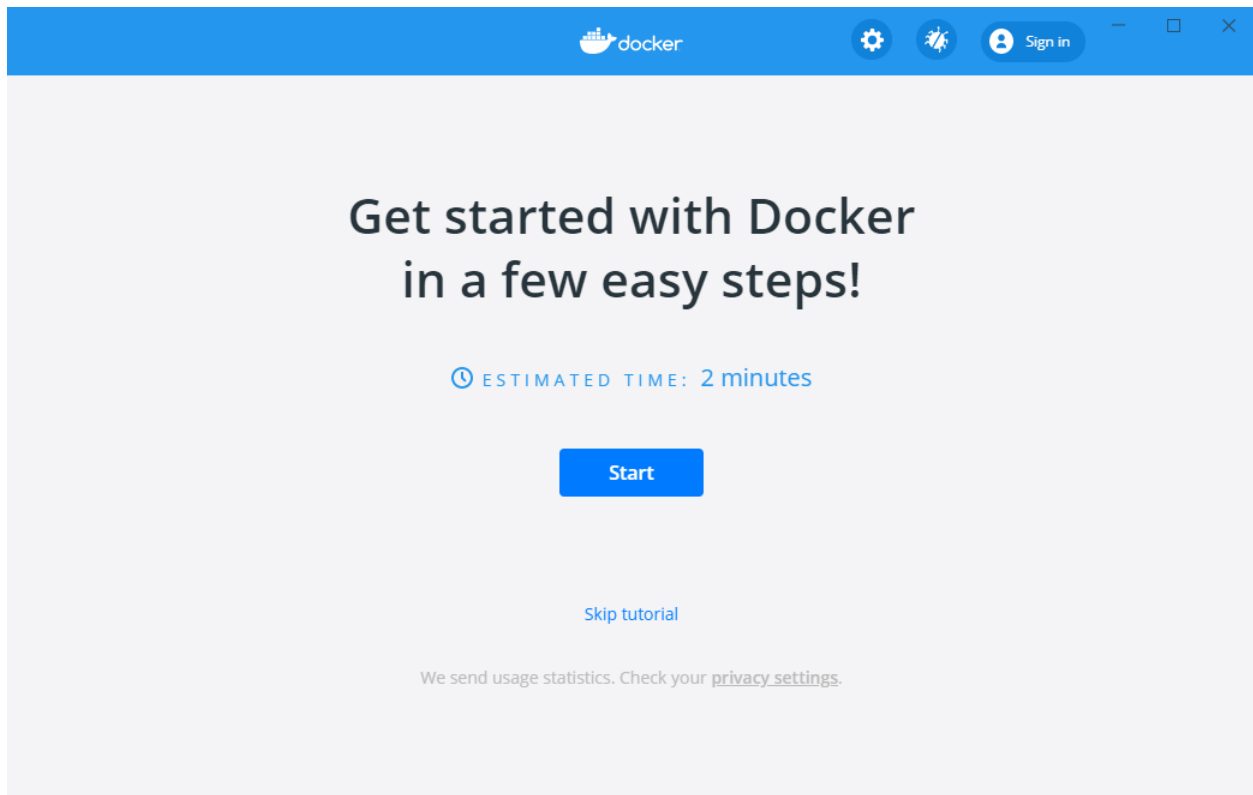
Docker Desktop does not start automatically after installation. To start Docker Desktop, search for Docker.



When the whale icon in the status bar stays steady, Docker Desktop is up-and-running.



Congrats! You are now successfully running Docker Desktop on Windows!



4.2 Install Docker on MacOS

Install Docker Desktop on Mac.

System Requirements:

- Docker Desktop - macOS must be version 10.13 or newer, i.e. High Sierra (10.13), Mojave (10.14) or Catalina (10.15).
- Mac hardware must be a 2010 or a newer model.

Installation:

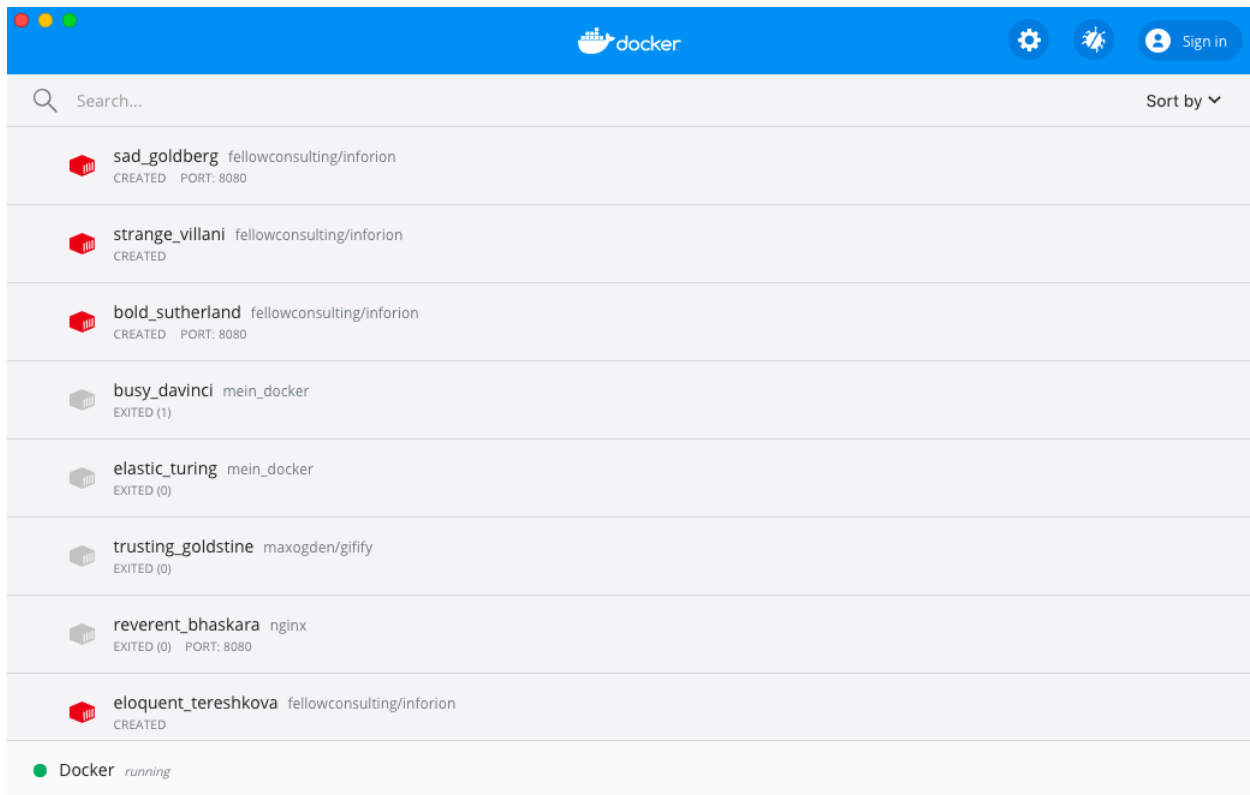
1. Double-click Docker.dmg to start the install process.
2. When the installation completes and Docker starts, the whale in the top status bar shows that Docker is running, and accessible from a terminal.



Start Docker Desktop:

Open a command-line terminal, and try out some Docker commands.

- Run `docker version` to check that you have the latest release installed.
- Run `docker run hello-world` to verify that Docker is pulling images and running as expected.



4.3 How to create flows on Prefect cloud using Inforflow Docker instance

```
docker run -v /local_path_to_conf_dir:/ion-workflow/conf fellowconsulting/inforion_
↳flow -b /ion-workflow/conf -c credentials.yml -m modules.csv -d module_dependencies.
↳csv
code . . .
```

- with **docker run** you can run your image as a container
- **-v** we are mounting our local directory to docker container, so it can use files from this directory
- with **flow** we are adding the configuration files that are missing inside the Docker ecosystem

If you see something similar to the screenshot below, it means that everything is running how it should be.

```
daniel.lopez@daniels-MacBook-Pro ~ % docker run -v /Users/daniel.lopez/conf:/ion-workflow/conf fellowconsulting/inforion flow -b /ion-workflow/conf -c credentials.yml -m modules.c
sv -d module_dependencies.csv
Login successful!
[{'module_name': 'parent_module'}]
[{'Supplier Orders': 'Suppliers'}]
[{'Supplier Order Lines': 'Supplier Orders'}]
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/3d56d178-9210-4aed-8822-b31b02750756
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/f6fc4608-fe32-46f0-9f07-ed76c35c4f5a
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/ea9408ef-744b-4081-9c5c-2eb5688a0d5b
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/772b2f61-458a-48c4-ba33-25aefde22e68

Prefect Agent

[2020-07-16 13:52:25.518] INFO - agent | Starting LocalAgent with labels ['2d4cc5528314', 'azure-flow-storage', 'gcs-flow-storage', 's3-flow-storage', 'github-flow-storage']
INFO:agent:Starting LocalAgent with labels ['2d4cc5528314', 'azure-flow-storage', 'gcs-flow-storage', 's3-flow-storage', 'github-flow-storage']
[2020-07-16 13:52:25.520] INFO - agent | Agent documentation can be found at https://docs.prefect.io/orchestration/
INFO:agent:Agent documentation can be found at https://docs.prefect.io/orchestration/
[2020-07-16 13:52:25.521] INFO - agent | Agent connecting to the Prefect API at https://api.prefect.io
INFO:agent:Agent connecting to the Prefect API at https://api.prefect.io
[2020-07-16 13:52:25.720] INFO - agent | Waiting for flow runs...
INFO:agent:Waiting for flow runs...
```

Copy and paste one of the links that appear after running the Docker in your browser.

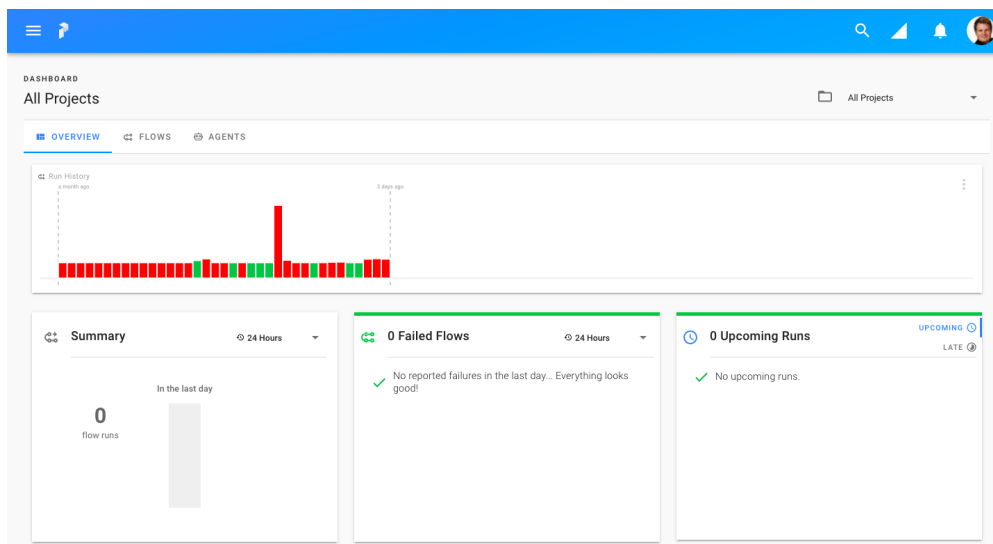
```
daniel.lopez@daniels-MacBook-Pro ~ % docker run -v /Users/daniel.lopez/conf:/ion-workflow/conf fellowconsulting/inforion flow -b /ion-workflow/conf -c credentials.yml -m modules.c
sv -d module_dependencies.csv
Login successful!
[{'module_name': 'parent_module'}]
[{'Supplier Orders': 'Suppliers'}]
[{'Supplier Order Lines': 'Supplier Orders'}]
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/3d56d178-9210-4aed-8822-b31b02750756
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/f6fc4608-fe32-46f0-9f07-ed76c35c4f5a
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/ea9408ef-744b-4081-9c5c-2eb5688a0d5b
BVB
Result check: OK
Flow: https://cloud.prefect.io/fellow/flow/772b2f61-458a-48c4-ba33-25aefde22e68

Prefect Agent

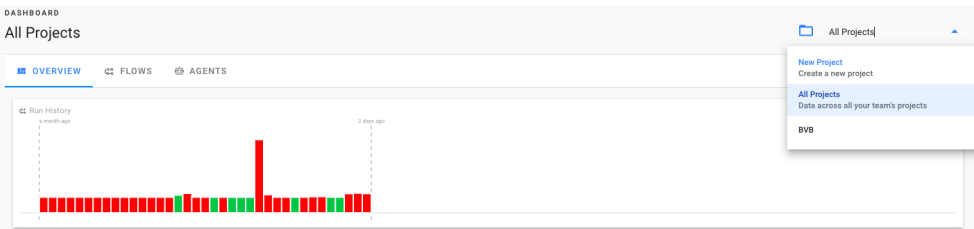
[2020-07-16 13:52:25.518] INFO - agent | Starting LocalAgent with labels ['2d4cc5528314', 'azure-flow-storage', 'gcs-flow-storage', 's3-flow-storage', 'github-flow-storage']
INFO:agent:Starting LocalAgent with labels ['2d4cc5528314', 'azure-flow-storage', 'gcs-flow-storage', 's3-flow-storage', 'github-flow-storage']
[2020-07-16 13:52:25.520] INFO - agent | Agent documentation can be found at https://docs.prefect.io/orchestration/
INFO:agent:Agent documentation can be found at https://docs.prefect.io/orchestration/
[2020-07-16 13:52:25.521] INFO - agent | Agent connecting to the Prefect API at https://api.prefect.io
INFO:agent:Agent connecting to the Prefect API at https://api.prefect.io
[2020-07-16 13:52:25.720] INFO - agent | Waiting for flow runs...
INFO:agent:Waiting for flow runs...
[2020-07-16 14:10:38.342] INFO - agent | Found 1 flow run(s) to submit for execution.
[2020-07-16 14:10:38.626] INFO - agent | Deploying flow run 7e68e6b3-af68-4ada-8c74-22e4a17e2b97
INFO:agent:Deploying flow run 7e68e6b3-af68-4ada-8c74-22e4a17e2b97
```

4.4 How to run flow from Prefect cloud

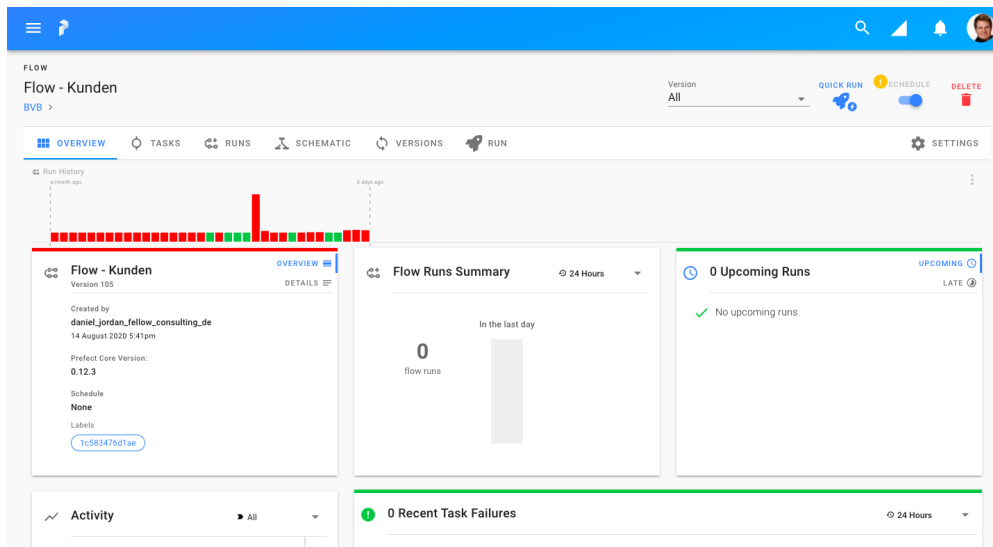
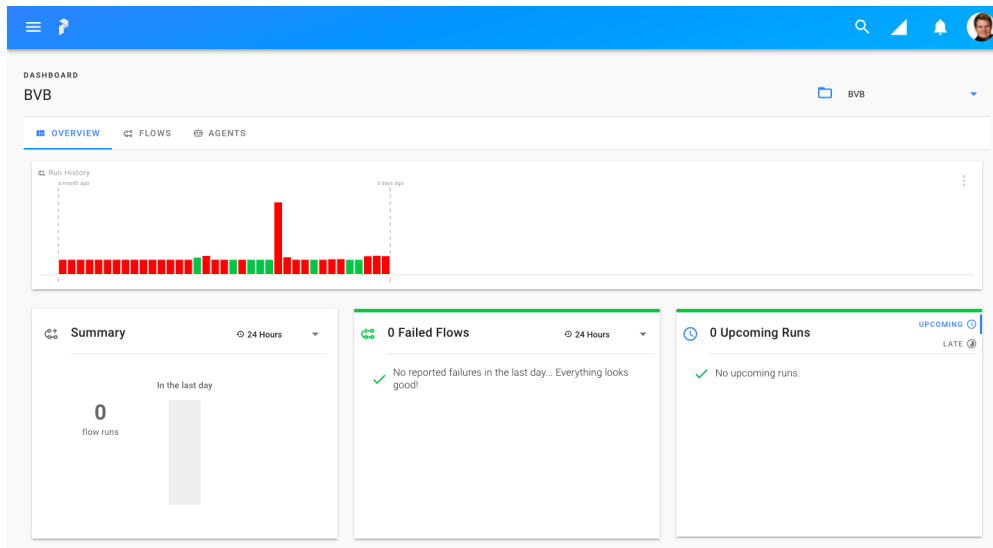
After clicking on the shown above link, you will be redirected to the Prefect main page - Dashboard. There you can manage all your existing projects.



From there you can go to right side of the window to select your project and to manage all its dataflows. In this case we are going to enter the **BVB** project.



Now, you have entered the BVB project and you can start running flows. In order to do so, just enter **Flows** to see the available and on-going flows and to put as an example, we are going to work with **Flow-Kunden**.



After selecting the flow, there is on top right corner a **quick run** button to run the flaw. It will be executed and the error that might occur will show up with a description to see in which exact task it went wrong.

FLOW

Flow - Kunden

BVB >

Version All

QUICK RUN

SCHEDULE

DELETE

OVERVIEW

TASKS

RUNS

SCHEMATIC

VERSIONS

RUN

SETTINGS

Run History

1 month ago

1 day ago

FLOW

Flow - Kunden

BVB >

Version All

QUICK RUN

SCHEDULE

DELETE

OVERVIEW

TASKS

RUNS

SCHEMATIC

VERSIONS

RUN

SETTINGS

Flow Runs

Search for a Flow Run

Name	Scheduled Start	Start Time	End Time	Duration	State
fuzzy-bumblebee	14 Aug 2020 5:42pm	14 Aug 2020 5:42pm	14 Aug 2020 5:42pm	25s	●
cordial-cassowary	14 Aug 2020 5:29pm	14 Aug 2020 5:30pm	14 Aug 2020 5:30pm	27s	●

FLOW RUN

fuzzy-bumblebee

BVB > Flow - Kunden >

RESTART

MASK AS

CANCEL

OVERVIEW

SCHEMATIC

GANTT CHART

LOGS

fuzzy-bumblebee

Version 4

Created by daniel_jordan_fellow_consulting_de

Last State Message [14 Aug 2020 5:42pm]: Some reference tasks failed.

Flow Run Version 14 August 2020 5:42pm Version 4

Scheduled Start Time 14 August 2020 5:42pm

Started 14 August 2020 5:42pm

Ended 14 August 2020 5:42pm

Duration 26 seconds

Activity

All

fuzzy-bumblebee Task Runs

Search by Task or Run Name

Task	Start Time	End Time	Duration	State
05. Kunden - Loading	14 Aug 2020 5:42pm	14 Aug 2020 5:...	4s	●
04. Kunden - Staging	14 Aug 2020 5:42pm	14 Aug 2020 5:...	2s	●
03. Kunden - Data Val...	14 Aug 2020 5:42pm	14 Aug 2020 5:...	10s	●
02. Kunden - Transfor...	14 Aug 2020 5:42pm	14 Aug 2020 5:...	3s	●
01. Kunden - Extraction	14 Aug 2020 5:42pm	14 Aug 2020 5:...	1s	●

Rows per page: 15 1-5 of 5 |< < > >|

CONFIGURATION FILES

InforFlow requires following 3 configuration files to run properly.

- Credentials (credentials.yml)
- Modules (modules.csv)
- Module Dependencies (module_dependencies.csv)

Credentials

Credentials file contains credentials for MS-SQL Server, M3 and Prefect Cloud.

- **MS-SQL Server**
 - Server name
 - Username
 - Password
 - Source Database name (Database from which data will be transferred to M3)
 - Staging Database name (Database where data will store after transformation)
- **Prefect cloud**
 - Project name
 - User token
 - Runner token
- **M3**
 - M3 URL
 - ION API File

m3:

```
url: {{M3_URL}}
ionfile: {{ION_API_FILE_NAME}}
```

database:

```
servername: {{SERVER_NAME}}
username: {{USER_NAME}}
password: {{PASSWORD}}
db_extraction: {{DB_EXTRACTION}}
db_staging: {{DB_STAGING}}
```

prefect:

```
project_name: {{PREFECT_PROJECT_NAME}}
user_token: {{USER_TOKEN}}
runner_token: {{RUNNER_TOKEN}}
```

All these credential informations should be provided in yaml format and file name should be provided in parameter when creating flow.

Modules information

For every module you want to export data from MS-SQL Server to M3 you need following information and add this to a csv file.

- Module name (You can put any name here. This is only used to create flows)
- Program name (This should be valid M3 program name where data will be exported)
- Mapping excel file path (Full path for the excel file which contains the sheet used for data transformation)
- Sheet name (Name of the sheet which contains all the transformation information)
- Source table name: Table name from which data will be extracted.
- Stage table name: Table name where data will be stored after transformation.
- Start: Start row for extraction [inclusive]. 1 means first row.

- End: End row for extraction [inclusive]. -1 means all records starting from start row. (Hint: This is not number of rows instead end row. e.g Start 2, End 6 means Rows 2, 3, 4, 5 and 6).

```
module_name, program_name, methods, mapping_file, sheet_name, source_table_name, stage_table_name, start, end
Kunden, CRS610MI, Add, BVB_Mapping_Kundenstamm.xlsx, Kunden, [dbo].[Customer_master_crs610], CRS610, 0, -1
```

All this information should be provided in csv format and file name should be provided in parameter when creating flow.

Modules dependencies information

This file contains dependencies informations. Like if a module is dependent on another module and should run only after parent module is migrated. e.g. OrderLineItems is dependent on Orders and should be transferred after Orders are transferred.

```
module_name, parent_module
Supplier Orders, Suppliers
Supplier Order Lines, Supplier Orders
```

This information should be provided in csv file in a parent-child format and file name should be provided in parameter when creating flow.

GREAT EXPECTATIONS INTEGRATION

Infor Flow allows data validation using Great Expectations. Great Expectations uses a unique tool to automated testing, also known as “pipeline tests”, which focuses on giving teams an analytical integrity and on helping with the management of complex codebases. Pipeline tests are like unit tests for datasets: they help you protect from upstream data changes and monitor data quality. Essentially it accelerates the ETL and data normalization process and automates verification of new data deliveries.

6.1 Creating Expectation Suites

To create expectation suites, user need to pass following 3 parameters

- Root directory for configurations
- Filename for YAML file containing credentials. e.g. credentials.yml
- Filename for CSV file containing modules information. e.g. modules.csv

```
docker run -v /local_path_to_conf_dir:/ion-workflow/conf -v /local_path_to_great_
↳expectations_dir:/ion-workflow/great_expectations fellowconsulting/inforion great_
↳expectation create -b /ion-workflow/conf -c credentials.yml -m modules.csv
```

- with **docker run** you can run your image as a container
- **-v** we are mounting our local directory to docker container, so it can use files from this directory
- with **create** we are telling the library to create great expectation suites

6.2 List Expectation Suites

To view all great expectation suites

```
docker run -v /local_path_to_conf_dir:/ion-workflow/conf -v /local_path_to_great_
↳expectations_dir:/ion-workflow/great_expectations fellowconsulting/inforion great_
↳expectation list -b /ion-workflow/conf -c credentials.yml
```

6.3 Update Expectation Suites

To update great expectation suites

```
docker run -it -p 8080:8080 -p 8888:8888 -v /local_path_to_conf_dir:/ion-workflow/  
↪conf -v /local_path_to_great_expectations_dir:/ion-workflow/great_expectations_  
↪fellowconsulting/inforion great_expectation update -b /ion-workflow/conf -c_  
↪credentials.yml -s [suite_name]
```

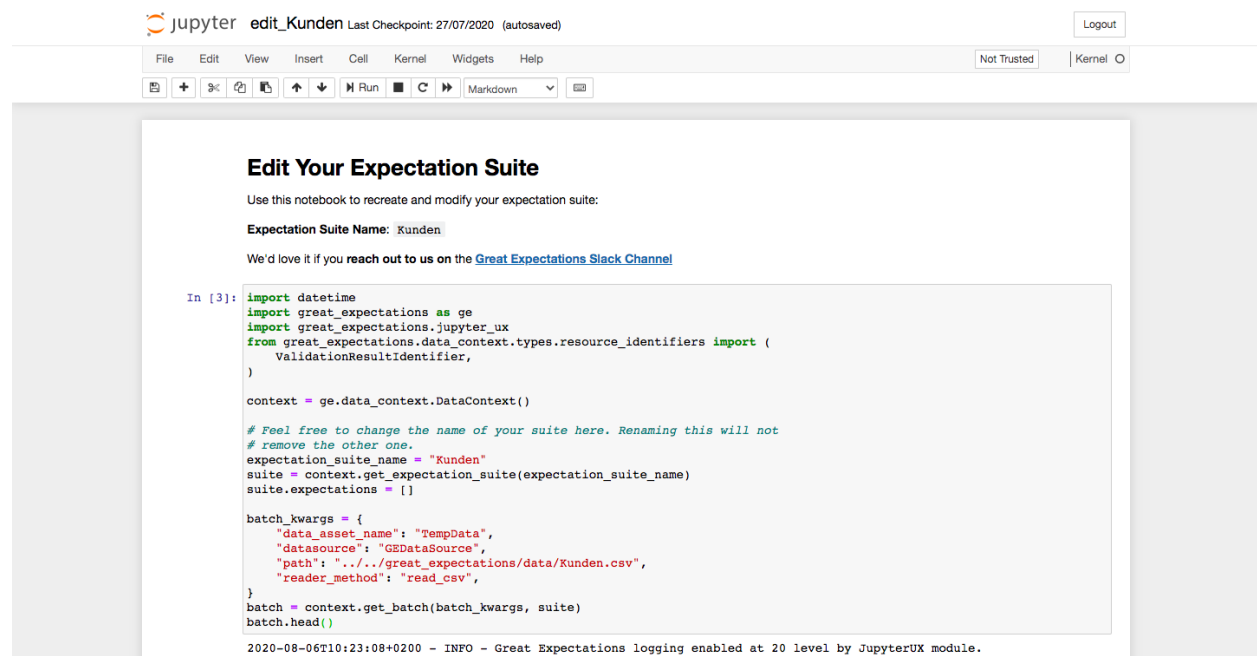
- with **s** we are telling which suite we want to update

GREAT EXPECTATIONS EXAMPLE

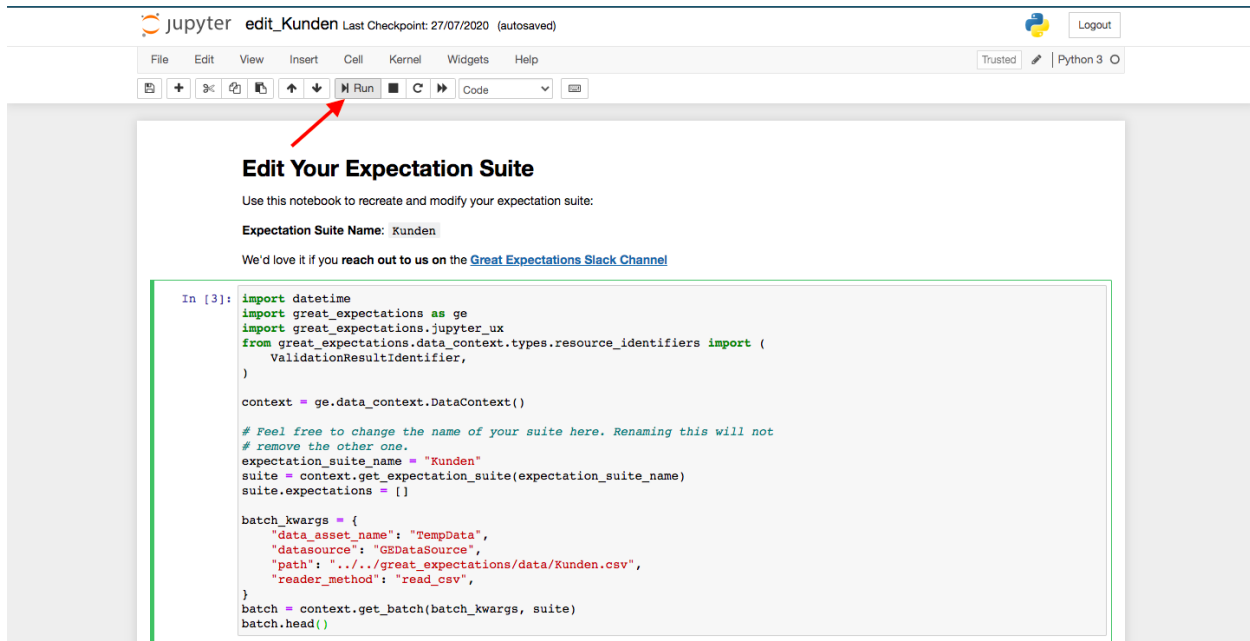
You can access Great Expectations by copying the last link provided by the outcome command (*Update Expectation Suites*). You should enter this link in your web browser to be able to access **Jupyter Notebook**, an open-source web application. If you have entered as suite name, Kunden for example, you should get something like this:



Now you can edit any of the shown above suites expectations. In this case, as an example, we are going to edit “edit_Kunden.ipynb”.



First of all, we are going to run the first input command to see the output data we get. Furthermore, it is an important first step, so you can get started with your customized creation and edit of new expectations. All you have to do is click on the upper part of the Jupyter Notebook, and run the first input, *as shown below with the arrow*.



The screenshot shows a Jupyter Notebook titled 'edit_Kunden'. The 'Run' button in the toolbar is highlighted with a red arrow. The notebook content includes a title 'Edit Your Expectation Suite', instructions to use the notebook to recreate and modify an expectation suite, and a code cell with the following Python code:

```
In [3]: import datetime
import great_expectations as ge
import great_expectations.jupyter_ux
from great_expectations.data_context.types.resource_identifiers import (
    ValidationResultIdentifier,
)

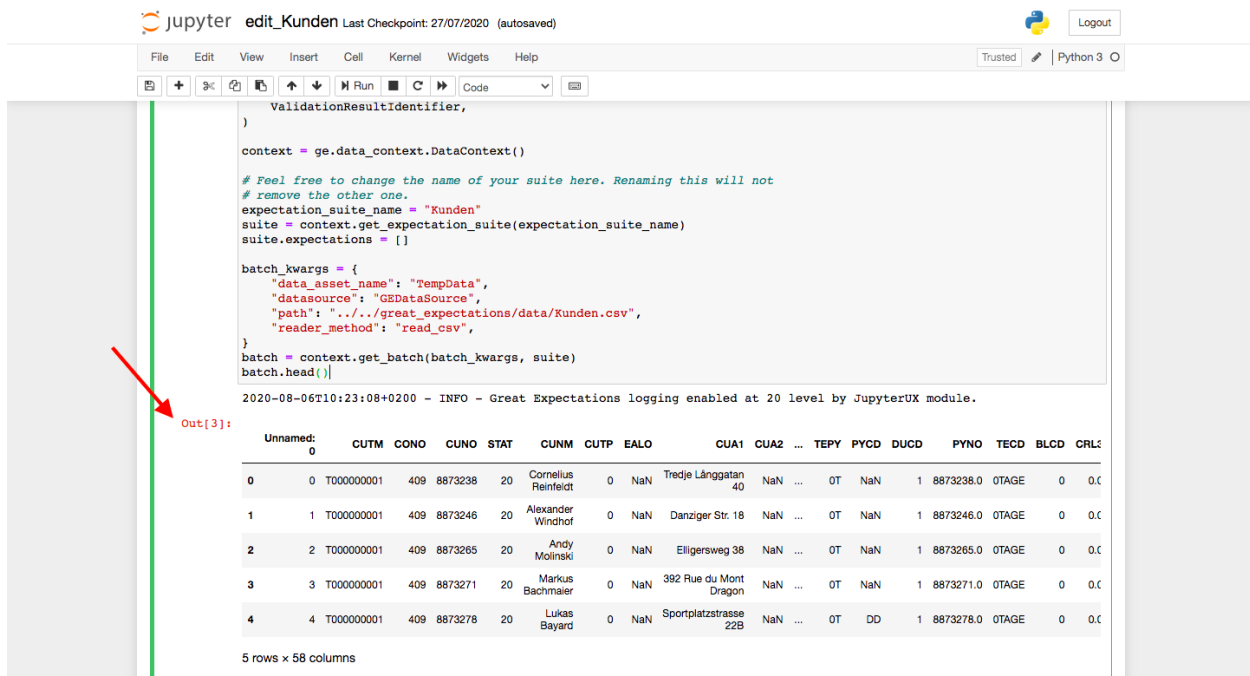
context = ge.data_context.DataContext()

# Feel free to change the name of your suite here. Renaming this will not
# remove the other one.
expectation_suite_name = "Kunden"
suite = context.get_expectation_suite(expectation_suite_name)
suite.expectations = []

batch_kwargs = {
    "data_asset_name": "TempData",
    "datasource": "GSDatasource",
    "path": "../../great_expectations/data/Kunden.csv",
    "reader_method": "read_csv",
}

batch = context.get_batch(batch_kwargs, suite)
batch.head()
```

So the output that we get is the following one:



The screenshot shows the same Jupyter Notebook, but now the output of the code cell is visible. A red arrow points to the output. The output is a table with 58 columns and 5 rows. The first row is the header, and the following rows are data. The table is titled 'Out[3]:'.

Unnamed: 0	CUTM	CONO	CUNO	STAT	CUNM	CUTP	EALO	CUA1	CUA2	...	TEPY	PYCD	DUCD	PYNO	TECD	BLCD	CRL2	
0	0	T000000001	409	8873238	20	Cornelius Reinfeidt	0	NaN	Tredje Långgatan 40	NaN	...	OT	NaN	1	8873238.0	OTAGE	0	0.0
1	1	T000000001	409	8873246	20	Alexander Windhof	0	NaN	Danziger Str. 18	NaN	...	OT	NaN	1	8873246.0	OTAGE	0	0.0
2	2	T000000001	409	8873265	20	Andy Molinski	0	NaN	Elligersweg 38	NaN	...	OT	NaN	1	8873265.0	OTAGE	0	0.0
3	3	T000000001	409	8873271	20	Markus Bachmaier	0	NaN	392 Rue du Mont Dragon	NaN	...	OT	NaN	1	8873271.0	OTAGE	0	0.0
4	4	T000000001	409	8873278	20	Lukas Bayard	0	NaN	Sportplatzstrasse 22B	NaN	...	OT	DD	1	8873278.0	OTAGE	0	0.0

5 rows x 58 columns

Now we can start creating and editing our desired expectations. Moving down to the next steps, we enter **Create & Edit Expectations**. It is important to know, that for any doubts or further explanations you can always check out the [expectation glossary](#) Nevertheless, you can see that you can put your new or modified expectation requirements in **Table expectations** (for tables) or **Column Expectations** (for each column).

Create & Edit Expectations

Add expectations by calling specific expectation methods on the `batch` object. They all begin with `.expect_` which makes autocompleting easy using tab.

You can see all the available expectations in the [expectation glossary](#).

Table Expectation(s)

```
In [ ]: batch.expect_table_column_count_to_equal(value=2)
```

Column Expectation(s)

CUNO

```
In [ ]: batch.expect_column_values_to_be_unique("CUNO")
```

As an example, we are going to run a new *expectation* for **Column Expectations**, and what we want to do is to put a requirement for the column named “CUNO”, so that all values are unique. You can always check out the existing columns by coming back to the first output, where your working data is.

Table Expectation(s)

```
In [2]: batch.expect_table_column_count_to_equal(value=2)
```

```
Out[2]: {
  "exception_info": null,
  "result": {
    "observed_value": 58
  },
  "meta": {},
  "success": false
}
```

Column Expectation(s)

CUNO

```
In [3]: batch.expect_column_values_to_be_unique("CUNO")
```

```
Out[3]: {
  "exception_info": null,
  "result": {
    "element_count": 200,
    "missing_count": 0,
    "missing_percent": 0.0,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "unexpected_percent_nonmissing": 0.0,
    "partial_unexpected_list": []
  },
  "meta": {},
  "success": true
}
```

JupyterLab interface showing a code cell with Great Expectations configuration. The code defines a suite named 'Kunden' and a batch. The output is a table with 58 columns. A red arrow points to the 'CUNO' column in the table.

```

)
ValidationResultIdentifier,
)

context = ge.data_context.DataContext()

# Feel free to change the name of your suite here. Renaming this will not
# remove the other one.
expectation_suite_name = "Kunden"
suite = context.get_expectation_suite(expectation_suite_name)
suite.expectations = []

batch_kwargs = {
    "data_asset_name": "TempData",
    "datasource": "GSDatasource",
    "path": "../great_expectations/data/Kunden.csv",
    "reader_method": "read_csv",
}

batch = context.get_batch(batch_kwargs, suite)
batch.head()

```

2020-08-06T10:23:08+0200 - INFO - Great Expectations logging enabled at 20 level by JupyterUX module.

Out[3]:

Unnamed: 0	CUTM	CONO	CUNO	STAT	CUNM	CUTP	EALO	CUA1	CUA2	...	TEPY	PYGD	DUCD	PYNO	TEGD	BLGD	CRL2	
0	0	T000000001	409	8873238	20	Cornelius Reinfeidt	0	NaN	Tredje Långgatan 40	NaN	...	OT	NaN	1	8873238.0	OTAGE	0	0.0
1	1	T000000001	409	8873246	20	Alexander Windhof	0	NaN	Danziger Str. 18	NaN	...	OT	NaN	1	8873246.0	OTAGE	0	0.0
2	2	T000000001	409	8873265	20	Andy Molinski	0	NaN	Elligersweg 38	NaN	...	OT	NaN	1	8873265.0	OTAGE	0	0.0
3	3	T000000001	409	8873271	20	Markus Bachmaier	0	NaN	392 Rue du Mont Dragon	NaN	...	OT	NaN	1	8873271.0	OTAGE	0	0.0
4	4	T000000001	409	8873278	20	Lukas Bayard	0	NaN	Sportplatzstrasse 22B	NaN	...	OT	DD	1	8873278.0	OTAGE	0	0.0

5 rows x 58 columns

Finally, in case you want to add more expectations (new cells), whether for tables or for columns, just be sure to be in the input row of *Table expectations*, for example. Then you can choose in the upper part of jupyter to **insert a new cell below or above**.

JupyterLab interface showing the 'Insert Cell Below' menu option. The code cell contains a Great Expectations expectation.

```

In [2]: batch.expect_table_column_count_to_equal(value=2)

```

Out[2]:

```

{
  "exception_info": null,
  "result": {
    "observed_value": 58
  },
  "meta": {},
  "success": false
}

```

JupyterLab interface showing a new code cell being added below the previous one.

```

In [ ]:

```


COMMON ERROR MESSAGES

Error 1

Generic exception. Firstly it will give you a string. If the error. Please contact the developer.

Error 2

The file file was not found or does not exist. Please check it.

Error 3

Credentials or module file is empty. Please make sure to have the proper information in the file.

Error 4

One of the column is empty in dependencies file. Please make sure to have the proper information in the file.

Error 5

The module name column must be unique. Each module can be dependent on a single module. Please make sure that the module name is only used once.

Error 6

The introduced URL is invalid. Please check and give a valid URL.

Error 7

Parameters in the file were not found.

Error 8

Required Column is missing.

Error 9

The input was not a valid integer.

Error 10

Path provided for base directory does not exist.

INDICES AND TABLES

- `genindex`
- `modindex`
- `search`